

Logic Programming Engineering – Assignment 8

Dr.-Ing. Sascha Klüppelholz

Exercise 8.1 [Fibonacci numbers revisited]

- a) Declare a dynamic predicate `cachedfibonacci/2` (cf. `dynamic/1`) which serves to compute the n -th Fibonacci number. This time, the goal is to store the result of a computation of F_n in memory and make it available for future computations to avoid costly re-computations.

To do so, write `cachedfibonacci/2` as wrapper that 1) calls the implementation of the predicate `fibonacci/2` from Exercise 2.3a and 2) inserts the result to the cache via `asserta/1` on the dynamic predicate `cachedfibonacci/2`.

- b) Provide a modified version of the predicate `fibonacci/2` called `fastfibonacci/2` such that no Fibonacci number is computed more often than once.
- c) Evaluate the performance of the different implementations by using the SWI-profiler (cf. `profile/1`). Do not forget to empty the cache (cf. `retractall/1`) between two profiling runs.

Exercise 8.2 [Efficiency]

Download the Prolog solution files `eratosthenes.prolog`, `fibonacci.prolog`, and `nqueens.prolog` from the tutorials website. Extract the key ideas behind the different solutions and evaluate them by using the SWI-Prolog profiler (cf. `profile/1`).